논문스터디 2주차 1팀

고경현 주혜인 장이준

A Unified Approach to Interpreting Model Predictions

Scott M. Lundberg, Su-In Lee, “A Unified Approach to Interpreting Model Predictions”, University of Washington, NIPS 2017

읽기 전에,

- 여기서 말하는 모든 method는 모델의 해석을 위한 method를 말합니다. 이는 linear regression, logistic regression, decision tree와 같은 모델 그 자체로 결과를 설명할 수 있는 Interpretable model과 구분됩니다.
- Method는 크게 local method과 global method로 구분할 수 있습니다. Local method는 개별 예측값을 설명합니다. 간단히 말하면, 개별 예측값에 대해 변수 중요도를 구합니다. Global method는 모델의 전반적인 움직임(예측값의 평균)에 따라 변수 중요도를 구합니다.
- 이 논문은 Local method에 집중합니다.

참고: Chapter 8 Global Model-Agnostic Methods | Interpretable Machine Learning (christophm.github.io)

---

1. Introduction

모델이 왜 그렇게 예측했는지 설명하는 것은 예측하는 것만큼 중요하다. 이를 현대 모델에도 적용하기 위해 다양한 방법들이 등장했지만 여기엔 두 가지 문제가 있다.

1) 이 방법들이 어떤 연관성이 있는지 모른다.

2) 언제 어떤 방법을 사용하는 것이 좋은지 불분명하다.

저자는 이 두 문제에 대처하기 위해 결론적으로 새로운 모델 해석 framework인 **SHAP (Shapley Additive Explanations)**을 제안한다.

본 논문은 SHAP을 제안하는 과정에서 크게 세 가지 의의를 가지며, 이 세 가지 포인트가 논문에서 눈여겨 보아야 할 것들이다.

1. 모델 설명을 위한 방법들 간의 연관성을 찾기 위해 Additive feature attribution methods라는 개념 제시.
2. 이 새로운 개념에 해당하는 방법들 중 유일한 해를 보장하는 것은 Shapley value임을 증명하고, 이를 이용해 모델의 예측을 설명하는 새로운 방법인 SHAP을 제안.
3. 계산량이 많은 true SHAP value를 구하는 것 대신 다양한 방법을 이용해 approximation하는 방법을 제안하고, SHAP value가 다른 방법들에 비해 사람의 인식과 더 가까운 것을 밝힘.

위 세 가지 포인트들이 순서대로 등장한다.

2. Additive Feature Attribute Methods

해당 섹션은 두 가지 파트로 구분할 수 있다.

가장 먼저 기존 방법들 간 연관성을 설명 못하던 문제를 해결하기 위해 Additive feature attribute methods라는 틀을 새롭게 정의함으로써 기존 방법들을 하나의 범주로 묶고자 했다.

이후 기존 6가지의 방법이 이 개념에 해당함을 밝힘으로써 기존 방법들 간의 연관성을 설명한다.

Explanation model

우선 Additive feature attribute methods를 정의하기에 앞서 여기에 사용되는 개념인 explanation model을 알아야 한다.

Explanation model 역시 논문에서 정의된 개념으로, 복잡한 모델을 근사시켜 해당 모델의 결과를 설명할 수 있는 간단한 모델을 explanation model이라고 한다. 이 explanation model과 관련된 notation을 우선 보자.

- f : 설명이 필요한 원래 모델
- g : f를 설명하기 위한 단순화된 Explanation model
- x: f에 들어가는 original input
- x': g의 input으로 들어가는 x의 단순화된 형태
- h_x : x'을 x로 매핑하는 함수; $x = h_x(x')$
- z' : x'과 근접한 input value

라고 할 때, local method는 $g(z') \approx f(h_x(z'))$ 를 만족하고자 g를 모델링한다.

이게 무슨 말이냐면, Local method는 예측값 $f(x)$를 설명하기 위한 간단한 explanation model g를 찾고자 하는데, 이 g라는 모델은 f를 approximate ($g(z') \approx f(h_x(z'))$) 해야 한다는 것이다. 최소한 x'과 그 주변의 z'을 input으로 넣었을 때 output이 같아야(비슷해야) 단순화된 모델 g를 신뢰할 수 있을테니까.

▼ 공유하고 싶은 개인적인 의견..

> Explanation model이라는 개념이 굉장히 일반적이고 직관적인 개념인 것 같아서 인상 깊음... 나는 기존에 모델의 예측을 설명한다는 것은 변수별로 특정한 값(ex. feature importance, correlation coefficient)을 계산하는 것이라고 생각했는데, 그게 아니라 큰 틀에서 예측값 설명을 위한 “간단한” 모델을 새롭게 추정하는 것이니까,, 고정관념을 깨는 동시에 전반적인 방법론에 대한 직관적인 이해를 돕는 개념인 것 같아서 인상 깊고 좋은 것 같음.

이제 explanation model을 이해했으니, 기존의 방법을 하나의 틀로 규정하는 Additive feature attribute method를 정의할 차례다.

Definition 1.

> **Additive feature attribute methods**는 binary variable로 이루어진 선형의 explanation model g를 가진다. 즉 아래가 성립한다.

$$g(z') = \phi_0 + \sum_{i=1}^{M} \phi_i z_i', \tag{1}$$

$z' \in \{0,1\}^M$, M은 단순화된 input feature의 개수, $\phi_i \in \mathbb{R}$

여기에 포함되는 method들은 위와 같은 선형 explanation model를 가지며, 계수 ϕ_i가 바로 i번째 feature의 effect가 된다.

이제 기존 방법들이 여기 범주에 포함되는지, explanation model g를 갖는지 알아보자.

여기서는 방법론은 간단하게 개념적으로만 이해하고, 각 방법이 저자가 정의한 Additive feature attribution method에 속한다는 것만 알아둬도 좋다.

1. LIME (Local Interpretable Model-agnostic Explanations)

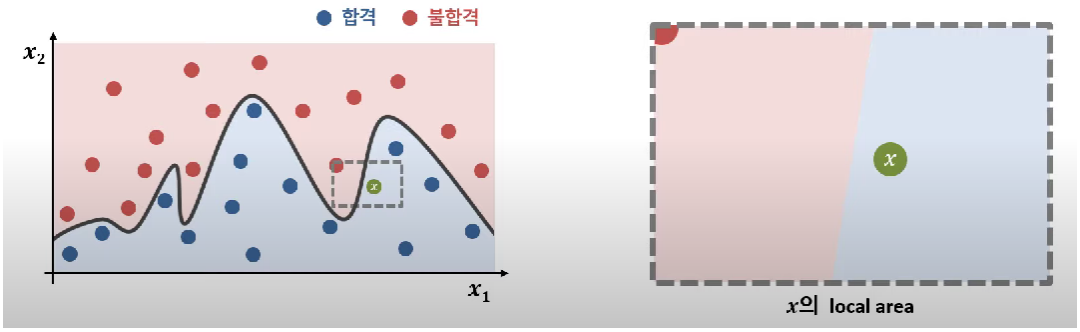
LIME은 개별 예측값이 왜 그렇게 예측되었는지 설명하는 대표적인 local method다. LIME은 다음과 같은 식을 만족하는 explanation model $\xi(x)$를 찾고자 한다.

$$\xi(x) = \text{argmin}_{g \in G} \ \mathcal{L}(f, g, \pi_{x'}) + \Omega(g) \tag{2}$$

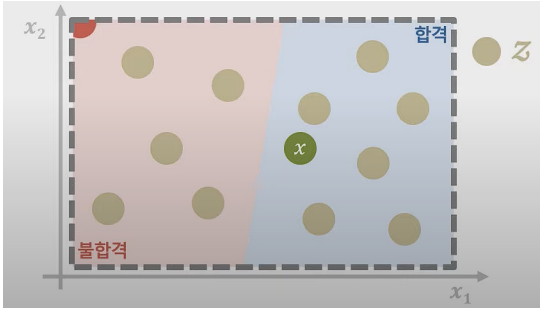
- f : 기존 complex한 모델
- $\mathcal{L}$: 손실 함수
- G : explanation model의 집합 (ex; linear model, decision tree)
- $\pi_{x'}$: 거리 기반 weight
- $\Omega(g)$: 찾고자 하는 explanation model이 복잡해지지 않도록 하는 penalty term

아래 그림을 통해서 쉽게 이해할 수 있다.

왼쪽 그림이 전체 데이터에 대한 complex model f라고 했을 때 LIME은 오른쪽 그림처럼 한 prediction에 변수가 얼마나 기여했는지 알기 위해 local area에서의 간단한 선형 모델 $\xi(x)$를 찾는다.



그것을 찾기 위해 아래 그림처럼 prediction 주변에 random하게 sample을 생성하여 (perturbed samples라고 표현함) locally 적용되는 linear model $\xi(x)$를 추정한다.



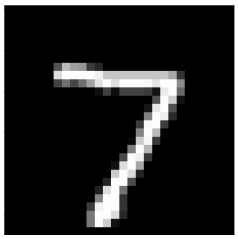
▼ 따라서 LIME이 찾고자 하는 것은 식 (1)에 있는 explanation model g과 일치하므로 LIME은 Additive feature attribute method다.

2. DeepLIFT

다음은 딥러닝의 예측 결과를 설명하는 method인 DeepLIFT다.

DeepLIFT는 사용자가 정하는 reference value라고 하는 대조군에 비해 우리가 갖는 original value(input)가 예측에 얼마나 기여하는지 계산한다.

이해하기 쉽도록 설명을 하기 위해 CNN으로 숫자 손글씨(MNIST) 분류를 하는 경우를 예로 들어보자.



위 그림을 CNN을 이용해 분류할 때 original value는 각 픽셀의 RGB값이다. reference value는 사용자가 정하는 값이므로 모든 픽셀이 검정색인 value라고 하자.

이때 7의 꺾인 부분(original value)이 모든 픽셀이 검정색인 부분(reference value)과 차이가 가장 많이나므로 해당 부분이 예측에 가장 많이 기여를 할 것이다. 이때 해당 부분의 DeepLIFT가 크게 나타나는 것이다.

이제 이 method가 Additive feature attribute methods인지 알아보자.

기여하는 정도를 수식으로 나타내면 다음과 같다.

$$\sum_{i=1}^{n} C_{\Delta x_i \Delta o} = \Delta o \tag{3}$$

- Δx_i : i번째 feature의 original value x_i와 reference value r_i의 차이

- Δo : original value와 reference value에 대응하는 output의 차이 ($\Delta o = f(x) - f(r)$)
- $C_{\Delta x_i \Delta o}$: Δx_i에 대응하는 기여도

▼ 여기서, $C_{\Delta x_i \Delta o} = \phi_i \ \& \ f(r) = \phi_0$이라고 하면 식 (1)과 일치한다. 따라서 이것 역시 Additive feature attribute methods이다.

3. Layer-Wise Relevance Propagation

▼ 이것 역시 딥러닝의 예측 결과를 설명하는 방법이다. DeepLIFT에서 reference가 특정한 수로 고정되는 경우라고 이해하자. 이 method도 Additive feature attribute methods이다.

4. Classic Shapley Value Estimation

게임 이론에 기반하여 feature importance를 구하는 *Shapley regression value, Shapley sampling values, Quantitative input influence* 세 가지 방법 역시 Additive feature attribute methods이다.

Shapley regression value는 다중공선성이 존재하는 선형 모델에서의 변수 중요도이다.

F를 전체 변수 집합, S는 F에서 변수 중요도를 알고 싶은 i번째 변수를 제외한 모든 변수들의 부분 집합이라고 할 때, i번째 변수의 Shapely regression value (feature importance) ϕ_i는 다음을 통해 계산할 수 있다.

$$\phi_i = \sum_{S \subseteq F \setminus \{i\}} \frac{|S|!(|F| - |S| - 1)!}{|F|!} [f_{S \cup \{i\}}(x_{S \cup \{i\}}) - f_S(x_S)] \tag{4}$$

위 식이 의미하는 바는 변수 i가 전체 모델의 예측에 기여하는 가중 평균값을 의미한다.

예를 들어 전체 변수가 A,B,C 3개 존재하고 변수 A의 Shapley value를 알고 싶을 때,

1. 변수 **A**를 넣었을 때의 output과 변수를 **아무것도 넣지 않았을** 때 output의 차이
2. 변수 **A,B**를 넣었을 때의 output과 변수 **B**를 넣었을 때 output의 차이
3. 변수 **A,C**를 넣었을 때의 output과 변수 **C**를 넣었을 때 output의 차이
4. 변수 **A,B,C**를 넣었을 때의 output과 변수 **B,C**를 넣었을 때 output의 차이

이 네 가지에 각각 가중치 $\frac{|S|!(|F|-|S|-1)!}{|F|!}$를 부여함으로써 변수 A가 전체 모델 f에서 기여하는 바를 계산한 것이 바로 Shapley value다.

변수 개수에 따른 가중치와 함께 모든 경우의 수를 고려하기 때문에 다중공선성이 존재함에도 활용할 수 있는 것이다.

▼ 이때 $\phi_0 = f_{\varnothing}(\varnothing)$ (변수를 넣지 않은 Null model)라고 한다면 Shapley regression value를 구하는 method 역시 linear explanation model g를 가지므로 Additive attribute methods이다.

언뜻 보면 알겠지만 Shapley regression value는 모든 변수 조합을 고려하기 때문에 계산이 매우 오래 걸린다. 이를 approximation하는 것이 Shapely sampling value나 Quantitative input influence라는 method인데, 이 역시

Additive feature attribution methods이다.

기존의 방법들이 Additive feature attribution methods라는 통일된 개념에 의해 연관성을 갖는 것을 알 수 있다. 그렇다면 이 중 어느 method를 사용해 모델을 설명하는 것이 좋을지 의문이다. 저자는 특정 정리(후술할 Theorem 1)에 의해 유일한 해를 보장하는 Shapley value가 이론적으로 좋은 method라고 말한다. 이후 Shapley value를 계산하는 framework인 SHAP을 제안한다.

3. Simple Properties Uniquely Determine Feature Attributions

위에서 설명한 additive feature attribution methods의 가장 놀라운 특성은 하단에 설명한 제약식들로 $\phi_i(f, x)$의 유일한 해들을 구해낼 수 있다는 점이다.

Property 1 (Local accuracy)

Local accuracy는 말 그대로 Explainable model인 $g(x')$가 주어진 local 데이터인 x에 대해서는 정확하게 예측해야 한다는 뜻이다. 다른 말로 풀어쓰자면, black box 모델인 $f(x)$의 local data x에 대한 output과, explainable model인 $g(x)$의 x'에 대한 output이 서로 동일해야 한다는 뜻이다.

$$f(x) = g(x') = \phi_0 + \sum_{i=1}^{M} \phi_i x'_i$$

Property 2 (Missingness)

Missingness는 simplified answer인 x'에서 feature i에 해당되는 값이 0이라면 ϕ_i 또한 0이 나와야 한다는 것을 뜻한다. 그리고 section 2에서 언급된 모든 methods(lime, deeplift) 등이 모두 이 조건을 따른다.

$$x'_i = 0 \Longrightarrow \phi_i = 0$$

Property 3 (Consistency)

Consistency는 만약 모델이 f에서 f'으로 바뀌었을 시 feature i번째 영향력이 더 커졌다면, $\phi_i(f', x)$값도 $\phi_i(f, x)$보다 커져야 한다는 뜻이다.

$$\begin{gathered} Let \quad f_x(z') = f(h_x(z')), \quad z'/i \quad is \quad z' \quad whose \quad z_i = 0 \\ f'_x(z') - f'_x(z'/i) \geq f_x(z') - f_x(z'/i), \\ then \quad \phi_i(f', x) \geq \phi_i(f, x) \end{gathered}$$

이 세 가지 property를 모두 만족하는 additive feature attribution methods의 solution $\phi_i(f, x)$는 하나의 unique한 값이 된다. 그것이 하단의 theorem 1에 해당된다.

Theorem 1

위의 세 가지 조건을 모두 만족하는 $\phi_i(f, x)$는 하단과 같다. (이를 증명하는 과정 자체는 굉장히 복잡하기 때문에 보충자료로 첨부해놓겠다.)

$$\phi_i(f, x) = \sum_{z' \subseteq x'} \frac{|z'|!(M - |z'| - 1)!}{M!}[f_x(z') - f_x(z'/i)]$$

- $|z'|$ is the number of the non-zero entries in z'
- $z' \subseteq x'$ represents all z' vectors where the non-zero entries are a subset of the non-zero entries in x'

어디서 많이 본 것같지 않은가? 맞다! SHAPLEY value이다.

결국 핵심은 세 가지 제약식을 모두 만족하는 additive feature attribution methods의 unique한 solution이 결국은 SHAPLEY value가 된다는 것이고 논문에서는 결국 SHAPLEY value가 XAI로 논리적인 지표임을 피력하고자 한 의도인 듯싶다.

4. SHAP (SHapley Additive exPlanation) Values

위에서 보았듯 SHAPLEY value의 정의는 하단과 같다.

$$\phi_i(f, x) = \sum_{z' \subseteq x'} \frac{|z'|!(M - |z'| - 1)!}{M!}[f_x(z') - f_x(z'/i)]$$

SHAP value는 SHAPLEY value 정의에서 $f_x(z')$ **대신** $E[f(z)|z_s]$**를 사용한다**는 차이밖에 없다.

$$\phi_i(f, x) = \sum_{z_s \subseteq x'} \frac{|z'|!(M - |z'| - 1)!}{M!}[E[f(z)|z_s] - E[f(z)|z_s/i]]$$

(참고로, $z' \subseteq x'$ 이고, x' 는 x의 부분집합에 해당되므로 z' 또한 본래 차원인 d차원보다 저차원인 형태라고 볼 수 있다.)

그렇다면 여기서 $f_x(z')$ = $f(h_x(z'))$ = $E[f(z)|z_s]$라는 값으로 근사하여 사용하는 이유는 무엇일까? 이는 실제 $f(h_x(z'))$ = $f(z_s)$를 구하는 과정을 찬찬히 생각해보면 답이 보일 것이다.

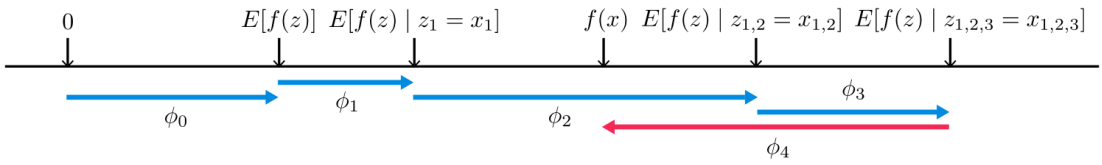
SHAPLEY value를 구할 때 $f_x(z')$ 를 알아야 하지만 우리가 현재 알고 있는 모델은 모든 input variable을 입력값으로 하는 $f(x)$라는 모델밖에 없다. (input 변수 종류의 변화에 따른 모든 $f_x(x')$를 구해놓진 않았기 때문이다.)

그래서 우리는 저차원의 z'을 원래 차원으로 다시 되돌릴 필요가 있고($f_x(z')$을 $f(h_x(z'))$로 변환), 이때 부분집합에 포함되지 않았던 variables(=missing values) 또한 다른 값으로 대체하여 재생성시킬 필요가 있다. 이 과정에서 missing value에 random한 값을 집어넣게 되는데, 이 random한 값의 arbitrary한 오류를 줄여주기 위해서 기댓값인 conditional expectation인 $E[f(z)|z_s]$을 사용하게 되는 것이다.

예시를 들어보자면, 타이타닉 생존자 예측 프로젝트에서 x의 원래 변수가 gender, age, sibiling만 있었고 이 세 가지 변수를 기반으로 만들어진 모델 $f(x)$가 있었다고 가정하자.

이때 z_s를 {age: 10, sibiling: 3}이라고 가정하고 $f(z_s)$를 구하고 싶다고 하자. 이때 {age: 10, sibiling: 3}만 입력값으로 넣으면 안된다. $f(x)$ 모델 자체는 gender, age, sibiling 총 세 개의 변수를 모두 입력값으로 받아야지만 도출되는 값이기 때문이다. 그래서 우리는 missing value인 gender를 임의의 값으로 대체해야 한다. (주로 0으로 대체한다고 한다) 하지만 이때 대체될 값이 패턴을 띄면 안되기 때문에 이를 상쇄시켜주기 위해 조건부 기댓값을 구한다고 생각하면 된다.

즉 SHAPLEY value에서 구해야하는 요소인 $f_x(z_s = \{age: 10, sibiling: 3\})$ 을 구하기 위해 $f(h'(z_s = \{age: 10, sibiling: 3\}))$ = $E[f(z)|z_s = \{age: 10, sibiling: 3\}]$ 로 근사시킨 후, marginal expectation인 $E_{gender}[f(z)]$을 구한다고 보면 된다.



상단의 다이아그램은 ϕ_i를 어떤 식으로 이해하면 되냐는 것을 도식적으로 보여준 것이다. 여기서는 single ordering을 가정하고 ϕ_i를 고려한 것이다.

여기서 ϕ_1가 대략 $[E[f(z)] - E[f(z)|z_1 = x_1]]$ 라는 값이라고 써놨는데, 사실 이 경우 말고도 $[E[f(z)|z_{1,2} = x_{1,2}] - E[f(z)|z_2 = x_2]]$, $[E[f(z)|z_{1,2,3} = x_{1,2,3}] - [E[f(z)|z_{2,3} = x_{2,3}]$, 등 x_1이 포함된 부분집합과 포함되지 않은 부분집합들을 모두 고려하여 그것들의 가중평균값으로 구해지는 값이긴 하다. (근데 single ordering이라고 가정했으니까! 헷갈리지 말 것!!)

이렇게 해서 SHAP value를 구하는 식이 등장하였지만 이것을 정확하게 계산하는 방법은 쉽지 않다고 한다. 하지만 현존하는 additive feature attribution methods를 결합하여 사용하게 되면 조금 더 편리한 방법으로 근사치를 구할 수 있게 된다.

model agnostic(모든 모델에 적용가능한)한 method로는 이전에 등장했던 Shapley sampling values, 그리고 새롭게 등장할 kernel SHAP이 있고, model specific(특정 모델에만 적용가능한)한 접근으로는 MAX SHAP 등이 존재한다.

이 방법들을 사용할 때 feature indepence와 model linearity를 가정하고(선택적이긴 한다.) 들어가는 것이 계산을 간단하게 해주는데 도움을 준다.

수리통계학입문(ft. BTS)의 내용을 조금씩 상기시켜보며 하단의 식을 보면 이해가 될 것이다. ‘

우선 $\overline{S}$은 S에 없는 feature을 담고있는 변수의 집합이라고 가정하자.

$$f(h_x(z')) = E[f(z)|z_s]$$

여기까지는 위의 설명으로 충분히 이해됐을 것이다.

$$\begin{aligned} f(h_x(z')) &= E[f(z)|z_s] \\ &= E_{z_{\overline{S}}|z_S}[f(z)] \end{aligned}$$

이때 z안의 모든 변수들이 independent하다고 가정하면 하단과 같이 나오게 된다.

$$\begin{aligned} &E_{z_{\overline{S}}|z_S}[f(z)] \\ &\approx E_{z_{\overline{S}}}[f(z)] \end{aligned}$$

모델의 선형성까지 가정하면

$$\begin{aligned} &E_{z_{\overline{S}}}[f(z)] \\ &\approx f([z_S, E[z_{\overline{S}}]) \end{aligned}$$

▼ 위의 식을 조금 더 자세하게

$$\begin{aligned} f(h_x(z')) &= E[f(z_s, z_{\overline{s}})|z_s] \\ &= E_{z_{\overline{s}}|z_s} E[f(z_s, z_{\overline{s}})|z_s, z_{\overline{s}}] \quad \dots \quad (1) \\ &= E_{z_{\overline{S}}|z_S}[f(z_s, z_{\overline{s}})] \\ &= E_{z_{\overline{S}}|z_S}[f(z)] \\ &\approx E_{z_{\overline{S}}}[f(z)] \\ &\approx f([z_S, E[z_{\overline{S}}]) \end{aligned}$$

equation (1)에서 수통입에서 배운 $E[X] = E[E[X|Y=y]]$ 개념이 쓰인 것이다.

4.1 Model-Agnostic Approximation

들어가기에 앞서 model-agnostic이란 모델의 종류와 관계없이 적용가능함을 의미한다.

조건부 기댓값을 추정할 때 feature independence를 가정하면 SHAP value들은 Shapley sampling values method와 Quantitative Input Influence method를 통해 추정될 수 있다. 더 적은 수의 input을 활용하여 계산하는 것이 합리적이지만, Kernel SHAP method는 비슷한 정확도의 추정을 하는데 더 적은 original model에 대한 evaluation을 요구한다.

Kernel SHAP (Linear LIME + Shapley values)

Linear LIME은 Linear한 설명 모델을 활용하는 LIME의 한 종류이며 이에 따라 additive feature attribution method이기 때문에 앞서 설명하였던 세 속성(local accuracy, missingness, consistency)을 만족할 때의 unique한 해가 Shapley value다. LIME의 목적함수는 $\xi(x) = \text{argmin}_{g \in G}\ \mathcal{L}(f, g, \pi_{x'}) + \Omega(g)$다. 이때, L, π'_x, Ω를 경험적으로 선택해야 하는데, 이런 방법을 사용하면 앞선 세 가지 속성을 만족하지 못하게 되는 등의 이유로 Shapley value를 발견할 수 없어지기에 경험적으로 선택하지 않고 Shapley value를 얻을 수 있는 L, π'_x, Ω를 찾는 방법을 소개한다.

Theorem 2 (*Shapley kernel*)

Under Definition 1 (Additive Feature Attribution Method), the specific forms of $\pi'_x,\ L,\ and\ \Omega$ *that make solution of Equation 2 (LIME objective function) consistent with Properties 1 through 3 are :*

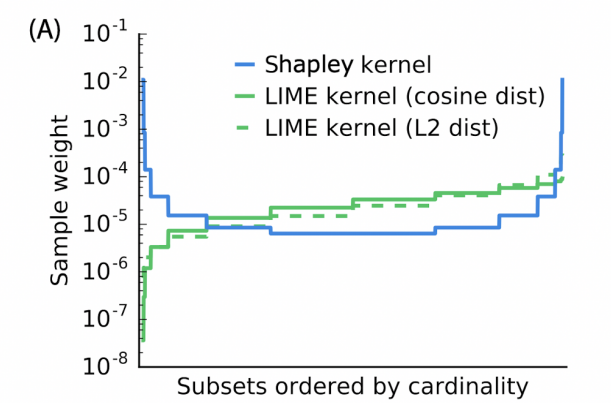
$$\Omega(g) = 0$$

$$\pi_{x'}(z') = \frac{(M-1)}{(M\ choose\ |z'|)\ |z'|\ (M - |z'|)}$$

$$L(f, g, \pi_{x'}) = \Sigma_{z' \in Z}\left[f(h_x^{-1}(z'))\ -\ g(z')\right]^2 \pi_{x'}(z')$$

$$where\ |z'|\ is\ the\ number\ of\ non-zero\ element\ in\ z'$$

Shapley value는 weighted linear regression을 활용하여 계산될 수 있다. LIME이 simplified input mapping을 사용하기에 SHAP value의 regression-based, model-agnostic estimation이 가능해진다.



이전의 경험적으로 선택되었던 kernel과는 다름을 보여준다. 모든 가능한 z'벡터들이 cardinality 순서대로 정렬되었을 때 Shapley kernel weighting은 대칭적인 모습이며, 이는 확연히 hueristic하게 선택된 kernel과는 다르다.

4.2 Model - Specific Approximations

Kernal SHAP은 model-agnostic이면서 속도가 느리다는 단점이 있는데, 아래의 방법들은 특정 모델에 제한되지만 더 빠른 approximation방법들이다.

Linear SHAP

linear한 모델들의 경우 input feature independence를 가정한다면, SHAP value들이 모델의 계수로부터 직접 추정될 수 있다.

corollary 1 (Linear SHAP)

$$Given\ a\ linear\ model\ f(x) = \Sigma_{j=1}^{M} w_j x_j\ +\ b:$$
$$\phi_0(f,x) = b\ and\ \phi_i(f,x) = w_j(x_j - E[x_j])$$

Low-Order SHAP

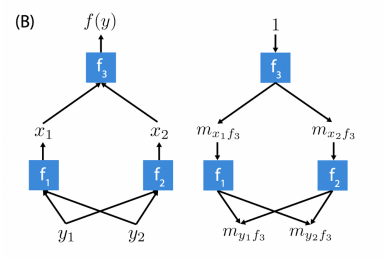
M이 작은 경우에 적절한 방법이다. 앞선 Theorem 2를 활용한 선형회귀는 $O(2^M + M^3)$의 복잡도를 지니기에 조건부 기댓값을 활용하여 approximation을 한다면 작은 값의 M이 효율적인 경우에 활용가능하다.

Max SHAP

최댓값을 증가시킬 수 있는 확률을 계산할 수 있는 방법이다.

Deep SHAP(DeepLIFT + Shapley values)

Kernel SHAP은 어떤 모델인지에 관계없이 사용가능한 반면, 성능을 향상시키기 위하여 deep network의 구조적인 특징을 활용하기 위하여 Shapley value와 DeepLIFT를 활용한 방법이다.



위의 neural network는 input이 $y = (y_1, y_2)$이며 은닉층 f_1, f_2과 output layer f_3을 거쳐 최종적으로 $f(y)$를 산출한다. 이러한 모델의 output을 1로 고정하고 1에 해당하는 output을 산출하기 위해 각 layer의 노드들이 어떤 contribution score를 갖는지 계산하는 것이 DeepLIFT였다.

DeepSHAP network의 일부분에서 계산된 SHAP value를 전체 network의 SHAP value로 결합한다.

f_1, f_2에서 산출된 x_1, x_2의 $f(y)$에 대한 contribution score$(\phi_i(f_3, x))$ 를 활용하는데, 이때 contribution score로 shapley value를 사용한다.

SHAP value는 만약 simple network component가 linear하다면 더 효율적으로 산출될 수 있는데 DeepSHAP는 각각의 component에서 계산된 SHAP value로부터 효율적인 선형화를 유도한다.

5. Computational and User Study Experiments

해당 논문에서는 computational efficiency, consistency with human intuition, explaining class difference 세 측면에서 SHAP이 우수한 성능을 보임을 나타냈다. dense decision tree와 sparse decision tree에서 Shapley sampling, SHAP과 LIME을 비교했을때, kernel SHAP의 샘플 효율성이 증가하였으며 SHAP은 다른 지표에 비하여 인간의 직관과 일치하는 경향을 보였다.

6. Conclusion

모델 예측의 정확도와 설명가능성에 대한 관심이 증가하면서, 예측을 해석할 수 있도록 하는 방법이 발전되었다. SHAP은 기존의 방법들 또한 additive feature importance methods임을 밝히고 필요한 특성을 만족할 경우 unique한 solution이 존재함을 설명하였다.

Appendix

그래프 해석하기

해당 예제는 shap package에 내장된 boston data를 xgboost로 예측한 후 이에 대한 shap value를 다양한 플랏으로 시각화한 과정을 나타낸 것입니다.

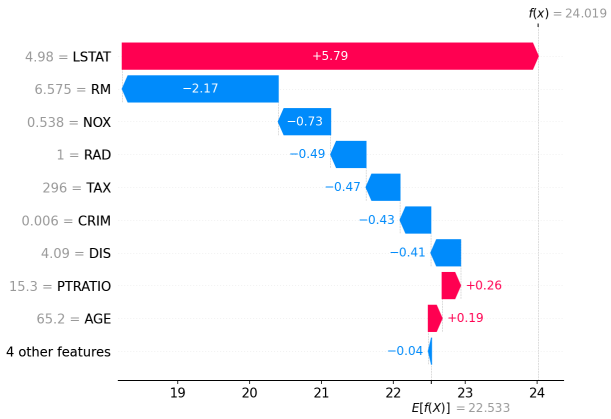
빨간색은 예측에 긍정적인 영향을, 파란색은 부정적인 영향을 의미합니다.

```
import xgboost
import shap

# train an XGBoost model
X, y = shap.datasets.boston()
model = xgboost.XGBRegressor().fit(X, y)

# explain the model's predictions using SHAP
# (same syntax works for LightGBM, CatBoost, scikit-learn, transformers, Spark, etc.)
explainer = shap.Explainer(model)
shap_values = explainer(X)
```

waterfall plot

아래의 waterfall plot은 가장 첫번째 예측값에 대한 플랏입니다.

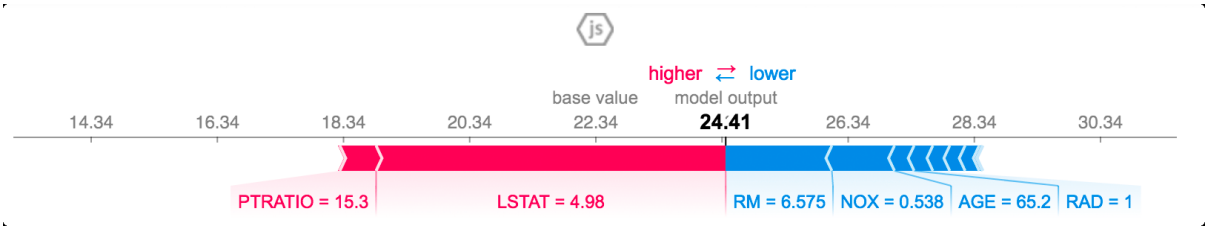
```
shap.plots.waterfall(shap_values[0])
```



Force plot

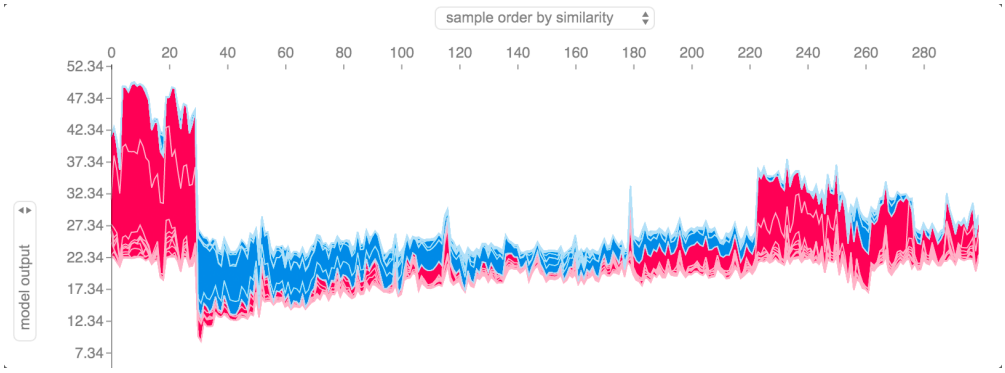
force plot은 데이터의 Shapley value를 1차원 평면에 정렬한 그래프입니다.

```
shap.plots.force(shap_values[0])
```



위의 force plot은 특정 데이터에 대한 shapley value를 표시한 플랏으로 첫번째 예측 값에 대한 force plot입니다.

```
shap.plots.force(shap_values)
```

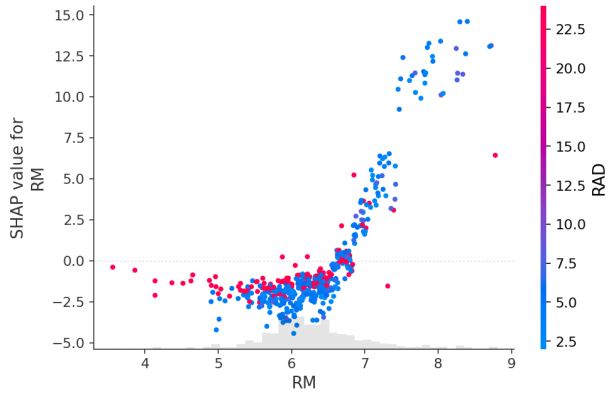


전체 데이터에 대해서도 Shapley value를 누적하여 확인할 수 있습니다.

scatter plot

하나의 feature가 어떻게 모델의 output에 영향을 미치는지에 대해 알아보고 싶을때 scatter plot을 통해 확인할 수 있습니다. 또한 동시에 오른쪽 y축인 RAD와의 interation effect도 확인할 수 있습니다.

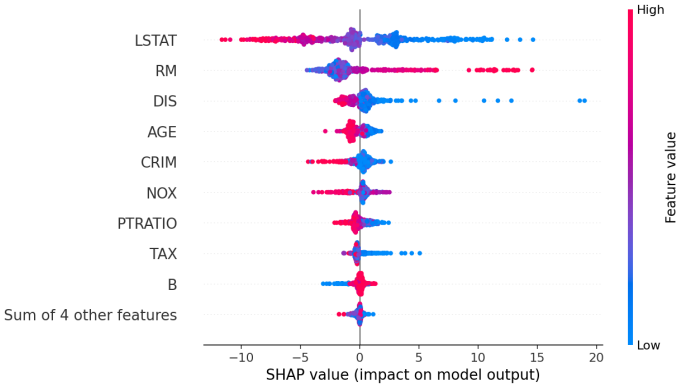
```
shap.plots.scatter(shap_values[:,"RM"], color=shap_values)
```



beeswarm plot

모든 feature들이 Shapley value의 분포에 어떤 영향을 미치는지를 보여주는 그래프입니다. 아래의 그래프를 해석해보면 LSTAT는 낮을수록 SHAP value가 높으며 RM은 높을수록 SHAP value도 높다고 볼 수 있습니다.

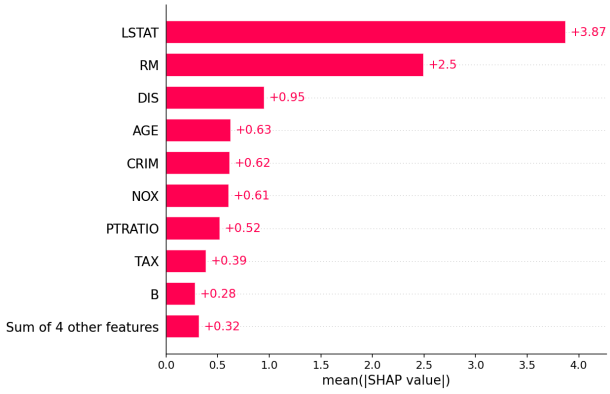
```
shap.plots.beeswarm(shap_values)
```



bar plot

아래의 bar plot은 각 feature별 SHAP value의 절댓값 평균을 시각화 한 그래프입니다.

```
shap.plots.bar(shap_values)
```



REFERENCE

https://ai.plainenglish.io/understanding-shap-for-interpretable-machine-learning-35e8639d03db

https://shap.readthedocs.io/en/latest/api_examples.html

https://github.com/slundberg/shap

▼ 참고

- 🍤 1 - 2
- 🌟 3 - 4.0
- 🐰 4.1-6